VŠB – Technická univerzita Ostrava

Fakulta elektrotechniky a informatiky

Bakalářská práce

2013 Jiří Gargaš

VŠB – Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Absolvování individuální odborné praxe
Individual Professional Practice in the company

2013

Jiří Gargaš

VŠB - Technická univerzita Ostrava
Fakulta elektrotechniky a informatiky
Katedra informatiky

Zadání bakalářské práce

Student: **Jiří Gargaš**

Studijní program: B2647 Informační a komunikační technologie

Studijní obor: 2612R025 Informatika a výpočetní technika

Téma: Absolvování individuální odborné praxe
Individual Professional Practice in the Company

Zásady pro vypracování:

1. Student vykoná individuální praxi ve firmě: ABB s.r.o.
2. Struktura závěrečné zprávy:
a) Popis odborného zaměření firmy, u které student vykonal odbornou praxi a popis pracovního zařazení studenta
b) Seznam úkolů zadaných studentovi v průběhu odborné praxe s vyjádřením jejich časové náročnosti
c) Zvolený postup řešení zadaných úkolů
d) Teoretické a praktické znalosti a dovednosti získané v průběhu studia uplatněné studentem v průběhu odborné praxe
e) Znalosti či dovednosti scházející studentovi v průběhu odborné praxe
f) Dosažené výsledky v průběhu odborné praxe a její celkové zhodnocení

Seznam doporučené odborné literatury:

Podle pokynů konzultanta, který vedl odbornou praxi studenta.

Formální náležitosti a rozsah bakalářské práce stanoví pokyny pro vypracování zveřejněné na webových stránkách fakulty.

Vedoucí bakalářské práce: **RNDr. Eliška Ochodková, Ph.D.**

Konzultant bakalářské práce: Ing. Ján Mináč

Datum zadání: 16.11.2012

Datum odevzdání: 07.05.2013

doc. Dr. Ing. Eduard Sojka
vedoucí katedry

prof. RNDr. Václav Snášel, CSc.
děkan fakulty

Prohlášení

Prohlašuji, že jsem tuto bakalářskou práci vypracoval samostatně.

Uvedl jsem všechny literární prameny a publikace, ze kterých jsem čerpal.

V Ostravě 2013

.....

Podpis

Poděkování

Rád bych na tomto místě poděkoval mému konzultantovi Ing. Jánovi Miná̌covi, za projevenou důvěru a příležitost k vykonání odborné praxe ve firmě ABB. RNDr. Elišce Ochodkové, Ph.D. za odborné vedení, za pomoc a připomínky, které přispěly ke zdárnému dokončení práce. Kateřině Dudkové za podporu, a všem kolegům ze softwarového oddělení za vřelé přijetí do kolektivu.

Abstrakt

Tato bakalářská práce se zabývá shrnutím odborné praxe ve firmě ABB, významné organizaci na poli energetiky a automatizace. Práce je rozdělena na tři části. První část představuje popis zaměření firmy a studentovo zařazení v ní. Druhá část řeší celkový popis informačního systému „Time Management Application“, přidělené úkoly a postup zpracování. Třetí část se věnuje dovednostem scházejícím, i těm, které byly touto praxí nabyty, závěrečným zhodnocením praxe a výsledkům dosaženým při vývoji požadovaného informačního systému.

Klíčová slova

ABB, Informační systém, Java, GWT, Databáze, JPA, Reporty, XML

Abstract

This bachelor thesis deals with the summary of professional experience in the company ABB, a major organization in the field of power and automation. The thesis is divided into three parts. The first part describes the focus of the company and student’s place in it. The second part deals with a general description of the information system "Time Management Application", assigned tasks and the processing procedure. The third part is about the skills which were missed and which were gradually gained by this practice, and contains final evaluation of the results achieved in the development of the required information system.

Keywords

ABB, Information System, Java, GWT, Database, JPA, Reports, XML

Seznam použitých zkratek a symbolů

AJAX	–	Asynchronous JavaScript and XML
API	–	Application Programming Interface
BIRT	–	Business Intelligence and Reporting Tools
CSS	–	Cascading Style Sheets
CZOPC	–	Operační centrum Česká republika
DB	–	Databáze
DTO	–	Data Transfer Object
DOM	–	Document Object Model
GWT	–	Google Web Toolkit
IDE	–	Integrated Development Environment
IS	–	Informační systém
JPA	–	Java Persistence API
QMS	–	Quality Management System
SAX	–	Simple API for XML
XML	–	Extensible Markup Language

Obsah

Seznam použitých obrázků

Seznam ukázek kódu

1 Úvod

Cílem této práce je popis praxe ve firmě ABB. Praxi jsem volil zejména z důvodů, že dnes firmy po absolventech ve většině případů požadují alespoň minimální praktické zkušenosti z oboru. Tímto jsem, doufám, alespoň minimálně zvýšil hodnotu v očích personalistů a šance na dobré pracovní uplatnění po zakončení studia.

První část práce velmi krátce popisuje firmu ABB, její odborné zaměření a mé pracovní zařazení v ní.

Hlavní část práce se zabývá popisem informačního systému „Time Management Application“, od jeho prvotního návrhu, až po přidělené úkoly na jeho implementaci. Tento IS zaštiťuje dvě hlavní funkce – správu pracovního času zaměstnanců a správu projektů, které firma vypracovává – a je pojat jako webový IS. Hlavním důvodem pro jeho vypracování byla snaha zmenšit papírovou administrativu.

Závěr je věnován znalostem a schopnostem, které jsem získal v průběhu studia, a které jsem v průběhu praxe využil. Dále těm, které mi scházely, a musel jsem je nabýt. A na konec dosaženým výsledkům a jejich zhodnocením.

Pro úplnost ještě dodám, že jsem IS vypracovával v týmu pod vedením Ing. Jána Mináče. Tato práce tak společně s prací Michaela Filsáka, dalšího studenta VŠB-TU Ostrava, který se věnoval implementaci druhé části IS, tvoří ucelený popis „Time Management Application“.

2 O společnosti ABB

ABB je přední světová skupina firem působících na poli energetických a automatizačních technologií, které pomáhají zlepšovat produkci a snižovat její dopad na životní prostředí. ABB, jako skupina firem, působí ve více než 100 zemích a zaměstnává okolo 145 000 lidí. Čistě v České republice působí tato skupina od roku 1970.

2.1 Popis odborného zaměření firmy

ABB poskytuje své služby prostřednictvím pěti hlavních divizí:

- Divize výrobků pro energetiku se zabývá výrobou a instalací energetických komponent (transformátory, rozvaděče a související zařízení) pro přenos a rozvod elektrické energie.

- Divize systémů pro energetiku poskytuje dodávky systémů pro přenosové a distribuční sítě a elektrárny. Také se zabývá instrumentací, řízením a elektrifikací elektráren.

- Divize automatizace výroby a pohonů vytváří výrobky, a s nimi související služby, pro zvýšení průmyslové produktivity a energetické účinnosti.

- Divize výrobků nízkého napětí vyrábí a distribuuje nízkonapěťové produkty, jako jsou jističe, spínače, montážní skříně nebo rozvaděče. Zároveň se soustředí na dodávku inteligentního systému řízení budov.

- Divize procesní automatizace poskytuje výrobky pro automatizace a optimalizaci průmyslových procesů. Součástí této divize je i CZOPC se sídlem v Ostravě [1].

Obrázek 1: Sídla ABB v České republice [2]

2.2 Popis pracovního zařazení studenta

ABB má v České republice pobočky v mnoha městech. V Ostravě se nachází CZOPC, které je součástí divize procesní automatizace. V rámci tohoto operačního centra působí několik oddělení. Jedním z nich je oddělení automatizace a softwarového vývoje s týmy aplikačních inženýrů, systémových inženýrů a softwarových vývojářů. Tato praxe byla absolvována na pozici softwarového vývojáře.

3 Zadané úkoly a jejich řešení

Hlavním zadaným úkolem, který byl plněn v průběhu celé praxe, bylo vytvoření webového IS „Time Management Application“. Tento systém řeší dva hlavní úkoly: správu pracovního času zaměstnanců a správu projektů. Hlavním důvodem, proč se začalo s vývojem, byla snaha umožnit zaměstnancům efektivní možnost, jak si spravovat pracovní dobu, dále umožnit jednoduchý přístup k základním informacím o projektech a zároveň možnost, jak zaznamenávat svou práci na daném projektu. Už z tohoto prvotního popisu bylo patrné rozdělení projektu do dvou hlavních modulů: modul pro správu pracovního času (dále v dokumentu jen jako Time modul) a modul pro správu projektů (dále v dokumentu jen jako modul QMS). V pozdější části vývoje IS se však ukázalo, že toto rozdělení není příliš vhodné, a proto byl projekt ve stávající formě rozdělen do jednotlivých, navzájem oddělených, komponent. Takto vznikl modul pro registraci, přihlášení, modul pro zobrazení kalendáře, modul pro správu kalendáře, modul manažerské správy, modul reportů a modul nastavení.

3.1 Výběr technologií a příprava prostředí

Vzhledem k tomu, že měl být daný IS vytvořen úplně od začátku, bylo nejprve nutné vybrat vhodné technologie pro vývoj a nastavit pracovní prostředí tak, aby mohla na vývoji IS spolupracovat celá skupina. Jako hlavní byly vybrány Java technologie propojené s frameworkem GWT, který umožňuje vytvářet AJAX aplikace psané čistě v jazyce Java. Výhoda GWT spočívá v tom, že se jedná o open source technologii. Na webu je tak k dispozici poměrně velké množství doplňkových komponent, vytvořených ostatními programátory, které mohou být dále volně využívány [3].

Jako hlavní technologie pro ukládání dat byla zvolena relační databáze Oracle Database 10g, z části proto, že jsou k této technologii dostupné i další technologie, které usnadňují práci s databází (např.: Oracle SQL Developer Data Modeler, který byl využit pro návrh DB) a z části také z důvodů předešlých zkušeností s prací s touto databází.

Protože práce na tomto IS probíhala ve skupině, bylo nejprve nutné zajistit správné verzování sytému. Nejprve bylo nutné nastudovat informace o dostupných verzovacích nástrojích, následně byl zvolen verzovací systém Mercurial, který se dá ve formě zásuvného modulu přidat do Eclipse IDE, ve kterém byl vyvíjen celý IS. Další výhodou zvoleného nástroje byl fakt, že existují již zavedené webové repositáře, které jej podporují a navíc doplňují o další funkcionalitu. V tomto případě se jednalo o podporu vytváření a přiřazování úkolů na projektu, která byla při vývoji IS použita [4] [5].

Celkově bylo tomuto úkolu věnováno přibližně prvních pět dnů, z nichž většinu času zabralo prozkoumání a příprava aplikace pro verzování systému a také framework GWT.

3.2 Návrh a implementace databáze

Tomuto projektu nebyl z bezpečnostních důvodů udělen přístup k žádné již zavedené databázi s údaji, které by bylo možné využít. Proto byla vytvořena vlastní oddělená databáze. Z hlavních požadavků projektu vyplývala potřeba udržovat data o zaměstnancích, jejich požadavcích ohledně pracovní doby, dále data o projektech a jednotlivých úkolech spojených s danými projekty. K dalším informacím, které bylo potřeba udržovat, patří například seznam typů úkolů, seznam platových center firmy nebo jazykové překlady. S ohledem na velikost dat se do DB dále ukládal pouze seznam typů úkolů. Z hlediska rozsahu by bylo ještě vhodné ukládat také jazykové překlady, ale pro ty byla zvolena technologie XML (viz kap. 3.6). Udržování ostatních stálejších dat bylo nakonec také vyřešeno pomocí XML. Následnou strukturu DB i s vazbami mezi jednotlivými tabulkami názorně ukazuje obrázek 2.

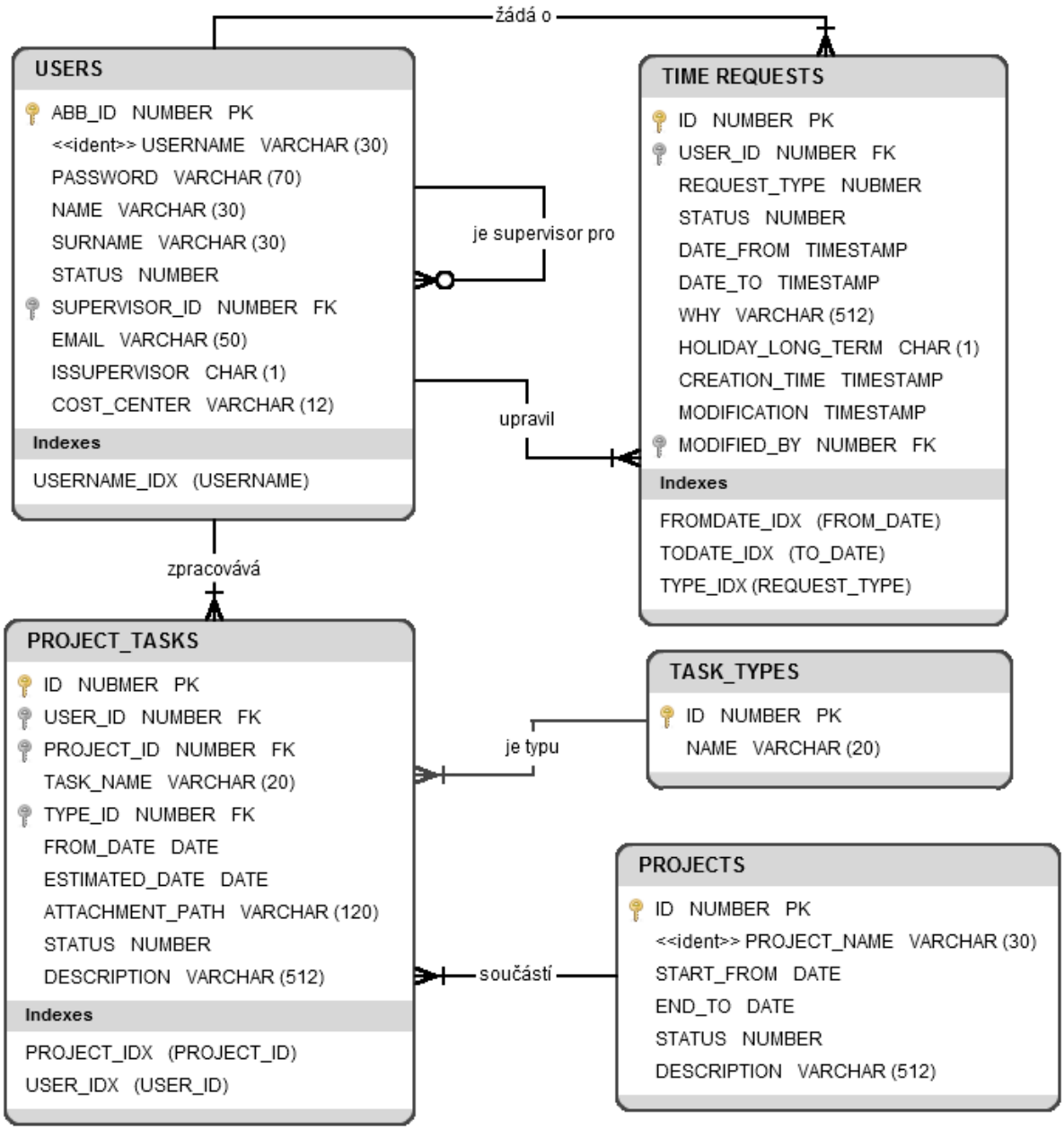
Po dokončení návrhu přišla na řadu samotná fyzická implementace databáze. Zde bylo, po úvaze a následné konzultaci o budoucích dotazech, rozhodnuto, že se pro všechny tabulky, vyjma záznamů o projektech, zvolí forma indexově orientovaných tabulek. Rozhodnutí bylo založeno na faktu, že u daných tabulek bude převládat vyhledávání a úprava údajů nad samotným vkládáním. Zbývající tabulka zůstala z opačného důvodu ve formě tabulky typu halda. Z úvahy o budoucích dotazech také vyplynula potřeba indexace nejčastěji používaných parametrů. Jako poslední byly vytvořeny sekvence pro primární klíče a triggery, které využívají tyto sekvence při každém vkládání do tabulek.

Tento úkol si celkově vyžádal přibližně tři dny práce. První dva dny byly věnovány návrhu a implementaci DB. Třetí den reprezentuje jednotlivé návraty k tomuto úkolu. Tyto byly způsobeny dodáváním atributů, na které se při návrhu DB zapomnělo, nebo se ukázaly jako potřebné v pozdějších fázích vývoje IS.

3.3 Propojení databáze s aplikací

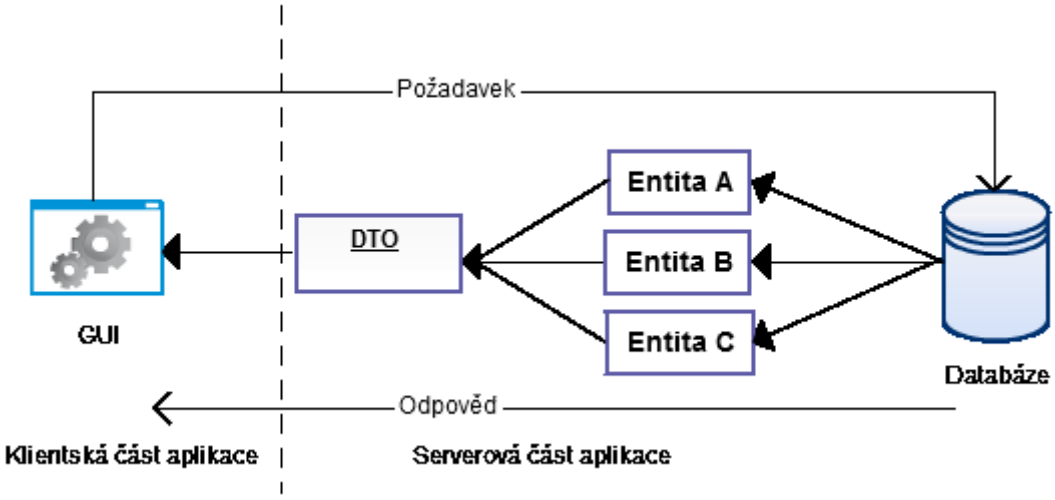
Jedním z dalších úkolů projektu bylo vyřešení způsobu posílání dat mezi databází, serverovou částí a klientskou částí aplikace. Tento úkol byl vyřešen objektově-relačním mapováním a využitím JPA. Jako implementaci tohoto API byl využit existující EclipseLink. Jeho použití vkládá mezi aplikační a datovou vrstvu další vrstvu, která zajišťuje komunikaci, a která by měla způsobit mírné zpomalení aplikace. Toto zpomalení zde nicméně pozorováno nebylo a použití JPA se tak v tomto případě jeví jako velmi výhodné.

Jelikož byla databáze nainstalována na serveru, kde byla spuštěna i samotná aplikace, vzniklo nejužší místo, z hlediska toku dat, mezi serverem a klientskou stanicí. Proto bylo potřeba zajistit vytváření korektních objektů se všemi potřebnými atributy tak, aby se klientovi po jediném dotazu vrátila všechna data, která by mohl následně upotřebit. Toto bylo vyřešeno použitím DTO. Takto vzniklé objekty v sobě udržují informace z více záznamů z jednotlivých tabulek. Toto řešení se ukázalo jako velmi výhodné, například při vkládání informací o projektech do kalendáře, kdy v sobě jednotlivé projekty udržují také informace o všech svých úkolech. Tím odpadá potřeba následné komunikace mezi klientskou a serverovou částí aplikace. Komunikaci přehledně zachycuje



Obrázek 2: Konceptuální model databáze

obrázek 3. Dále se tímto zajistila jistá filtrace posílaných dat. Ta byla potřebná, protože EclipseLink plní všechny atributy vytvářených entit, což není zcela žádoucí jev (např. při zaslání samotné entity uživatele do klientské části aplikace by po sítí putovalo společně s ostatními údaji i heslo uživatele) [6].



Obrázek 3: Tok dat z databáze do aplikace

Tento úkol si vyžádal tři dny. První dva dny probíhalo studování frameworku EclipseLink. Během třetího dne byla implementována většina funkcionality. K tomuto úkolu bylo nezbytné se ještě několikrát vrátit, z důvodů dopisování chybějících funkcí.

3.4 Modul registrace

I v tomto IS bylo třeba rozlišovat, kdo provedl kterou akci. Dalším úkolem proto bylo vytvoření registračního formuláře a následná implementace většiny jeho funkcionality. Důraz zde byl kladen na správné vyplnění formuláře. Také bylo důležité, z důvodů budoucí správy událostí, vytvořit správně organizační strukturu zaměstnanců. Tato struktura měla udržovat vztahy mezi zaměstnanci, tzn. kdo je přímý nadřízený daného zaměstnance, případně kdo všechno se danému zaměstnanci zodpovídá.

Správnost vkládaných údajů se ošetřovala jejich validací ještě před samotným odesláním údajů na server tak, aby nebylo zbytečně zatěžováno nejužší místo aplikace. Příklad použité validace názorně ukazuje zdrojový kód 1. V případě špatně vyplněných údajů systém zviditelnil upozornění skrytá za vstupními poli. V případě validace hesla byly použity regulární výrazy, ale zobrazení chyb nebylo prováděno pomocí zviditelnění varování. Kontrola shody obou polí pro zadání hesla probíhala kontinuálně. Na případné nesrovnalosti systém okamžitě upozornil podbarvením pozadí vstupních polí změnou použitého stylu z CSS.

V rámci vytvoření uživatele si také nový uživatel volil svého nadřízeného a dále platové středisko, do kterého spadá. Oba výběry byly vyřešeny pomocí číselníku. Pro získání nadřízených

pracovníků byla vytvořena funkce, která si potřebné údaje vybere z databáze. Naplnění číselníku platových center se však řešilo pomocí záznamů v konfiguračním XML souboru. Tato metoda byla zvolena zejména z důvodů malého počtu záznamů, a také pro možnost rychlé editace číselníku, bez potřeby přístupu do DB.

Následné vytvoření správné struktury zaměstnanců se původně mělo řešit přidáním tabulky do databáze. Vytvoření tabulky, která by mimo zaměstnanecké vztahy neudržovala žádné další informace, nebylo shledáno příliš efektivním. Byl zvolen efektivnější způsob, a to přidaní nového atributu entitě uživatele, pomocí kterého se odkazuje na přímého nadřízeného. Tato vazba je zobrazena i v obrázku 1 a je pojmenována „je supervisor pro“. Daný uživatel může mít pouze jednoho přímého nadřízeného, ale zároveň může být vedoucím pro mnoho dalších zaměstnanců. Tímto vznikla potřebná struktura přímo v tabulce uživatelů. Například při přihlášení, kdy se z entity uživatele vytváří DTO, které se mu následně vrací, tak nevzniká potřeba dalšího vyhledávání identifikačního čísla vedoucího pracovníka.

Tento úkol si vyžádal přibližně pět dnů, kdy samotné vytvoření formuláře zabralo zanedbatelnou část tohoto času. Větší část si vyžádalo řádné pochopení vytváření regulárních výrazů, které je náročné zapsat ve správné podobě.

```
private boolean isMailvalid ( String mainInput ) {
    String EMAIL_PATTERN =     "^[_A-Za-z0-9-\\+]+(\\.[_A-Za-z0-9-]+)*@[A-Za-z0-9-
    ]+(\\.[A-Za-z0-9]+)*(\\.[A-Za-z]{2,})$";

    return mainInput.matches(EMAIL_PATTERN);
}
```

Ukázka kódu 1: Kontrola zadaného e-mailu

3.5 Modul QMS

Dalším důležitým úkolem bylo vypracování modulu QMS. Ten se stará o veškeré požadavky spojené s firemními projekty. Z tohoto důvodu byl tento modul pojat jako kompozit částí, které zpracovávají primární funkce. Samotná funkcionalita modulu pak obsahuje vytváření jednotlivých projektů a možnost vytváření úkolů u vybraného projektu. Dále pak jejich jednoduché zobrazení a také možnost připojení souboru, a to jak k projektu, tak i k úkolům. Rozšiřující funkce pak zaštiťují změnu již vytvořených objektů.

Při samotném načtení modulu bylo zajištěno, že se v klientské části inicializovala kolekce projektů, a ta byla nadále udržována v paměti. Každý objekt si také v sobě udržuje kolekci úkolů. Načítáním celých kolekcí se minimalizovala komunikace mezi klientem a serverovou částí aplikace.

3.5.1 Vytváření a úprava projektů a jejich úkolů

Po vyvolání požadavku na vytvoření nového projektu se uživateli zobrazil formulář pro zadání všech potřebných informací. Součástí těchto informací je i doba, během které má být daný projekt realizován. Pro tyto data bylo potřeba zajistit korektní vyplnění. K tomu velmi dobře posloužila validace pomocí funkcí *before()* a *after()*, dostupných na instanci třídy Date [7].

Samotný požadavek na vytvoření nového projektu je rozdělen do několika částí. Jako první se provede vytvoření nového záznamu v databázi, následuje vytvoření složky projektu na pevném disku a jako poslední se vytváří informační soubor daného projektu. V případě, že by některá z těchto částí neproběhla úspěšně, jsou části jí předcházející smazány. V případě, že vše proběhne správně, vrátí se klientovi informace o provedení a následně se provede zapsání daného projektu do seznamu uloženého v paměti u klienta.

Pro následné vytváření a úpravu úkolů byly použity upravené funkce spojené s projektem. Vše tedy probíhá prakticky stejným způsobem.

3.5.2 Připojování souborů

K jednotlivým projektům a jejich úkolům bylo potřebné zavést také možnost nahrávání souboru, které k nim patří. To proto, aby se veškeré podklady nacházely na jednom místě a nebylo je nutné zdlouhavě dohledávat.

K tomuto byl využit již existující servlet, který umožňoval samotné nahrávání souborů na server. Co ale úplně neumožnil, byla specifikace cesty, kam se měl nahraný soubor uložit. Toto bylo vyřešeno zavedením proměnné v konfiguračním souboru aplikace. Tato proměnná v sobě udržuje cestu ke složce, ve které se vytvářejí všechny projekty. Tímto je také umožněna flexibilita výběru úložiště během nasazení aplikace. Následnou specifikaci konkrétní složky s projektem, a případně i s úkolem, byl zajištěn připojením jejich názvu k názvu ukládaného souboru. Na serveru se název parsuje a následně se tvoří finální cesta, do které se soubor uloží. Zdrojový kód 2 ukazuje část servletu, která zajišťuje samotné ukládání poslaných souborů [8].

Celkově vytvoření tohoto modulu zabralo přibližně osm dní. První polovinu této doby trvalo implementovat vytváření a úpravy projektů a úkolů, druhou polovinu pak dotvoření nahrávacího servletu tak, aby byl plně využitelný pro účely IS.

```
for (FileItem item : sentFiles) {
        if (false == item.isFormField()) {
                cont ++;
                // Create a new file based on the remote file name in the client
                try {
                        String saveName = item.getName()
                                .replaceAll("[\\\\/><\\|\\s\"'{}()\\[\\]]+", "_");
                // Parse target path
                        String separator = File.separator;
                        String filePath = item.getFieldName()
                                .replace("[:]", separator);
                // Create and fill file
                        File file = new File(filePath, saveName);
                        item.write(file);
                // Save a list with the received files
                        receivedFiles.put(item.getFieldName(), file);
                        receivedContentTypes.put(item.getFieldName(),
                                item.getContentType());
                // Send a customized message to the client.
                        response += "File saved as " + file.getAbsolutePath();
                } catch (Exception e) {
                        throw new UploadActionException(e);
        }
}
```

Ukázka kódu 2: Část nahrávacího servletu

3.6 Jazyková lokalizace

Předposlední úkol, který bylo nutné zpracovat, byla jazyková lokalizace. Ta byla potřebná, protože ABB je mezinárodní společností, tudíž firemním jazykem není čeština. Z tohoto důvodu byla aplikace primárně vyvíjena s anglickou lokalizací. Následně byla pro zvýšení uživatelského komfortu dodělána také lokalizace česká.

V GWT existuje možnost nastavení jazyka prostřednictvím nastavení proměnné „locale“ v konkrétním prohlížeči. Tato možnost byla shledána jako nevhodná, a proto byla uživatelům nabídnuto přepnutí jazyka přímo prostřednictvím rozhraní zobrazeného ve spodní liště IS. Tato lišta je přítomna v rámci celé aplikace a uživatel má tak možnost měnit jazyk kdykoliv to uzná za potřebné.

Jazyková lokalizace se začala ukládat do souboru XML, opět z důvodů snadného přístupu k uloženým datům, a také pro jednoduchý způsob editace. Připravený soubor s lokalizací se při spuštění aplikace na serveru parsuje na jednotlivé zprávy (objekt, který si udržuje unikátní

označení jak s českým, tak i anglickým překladem), které jsou uloženy v paměti. Tuto metodu ukazuje zdrojový kód 4. Samotné zpracování souboru probíhá pomocí DOM parseru, který si načte a následně zpracuje celý soubor. Tento parser je vhodné používat pouze pro menší soubory, u kterých funguje rychleji než SAX parser, který načítá jednotlivé záznamy. Z paměti se pak tyto zprávy, o které si uživatel při přístupu na jednotlivé stránky zažádá, načítají a následně se mu přeposílají. Nevzniká tak potřeba opakovaného parsování souboru [9].

Navíc byly jednotlivé metody, které pracují s jazykovou lokalizací, připraveny tak, že všechny změny, které je nutné udělat pro přidání dalšího jazyka, zahrnují pouze dva kroky. Prvním krokem je samotné přidání překladů do XML souboru. Druhým je vytvoření odkazu, který volá metodu změny použitého jazyka. Díky jednoduché a přehledné struktuře samotného XML souboru je zavedení nového jazyka pro překlad otázkou pár desítek minut.

```
<item>
        <message_key>loginModul-Username</message_key>
        <message language="en">Username:</message>
        <message language="cze">Uživatelské jméno:</message>
</item>
```

Ukázka kódu 3: Záznam v souboru jazykové lokalizace

Tento úkol si vyžádal celkově šest dní. Během nich byla vytvořena spodní lišta, do které se umísťují odkazy pro změnu jazyka. Většinu času ale zabralo studium dostupných metod parsování XML a následná úprava aplikace do podoby, ve které byly veškeré texty načítány z XML souboru.

3.7 Modul reportů

Posledním úkolem bylo zavedení reportů do IS. Samotné reporty slouží pro prezentování potřebných údajů ve vhodné a přehledné podobě. V tomto IS reporty posloužily pro možnost rychlého tisknutí žádostí o dovolenou a seznamu dnů, kdy konkrétní zaměstnanec pracoval z domova tak, aby tyto mohly být ve finální podobě poslány do administrativního centra ke zpracování.

Vytváření reportů lze v Java technologiích vyřešit pomocí existujících API. Po prvotním nezdaru s BIRT API se přistoupilo k použití JasperReports API. Příprava pro tvorbu reportů prostřednictvím tohoto API zahrnuje dva kroky. Prvním je vytvoření šablony reportu. K tomu lze využít mnoho dostupných nástrojů (např.: iReport Designer), nebo je možné šablony vytvářet přímo v libovolném textovém editoru. Samotná šablona není nic jiného než XML soubor se značkami, které určují jak statickou, tak dynamickou část souboru, datový zdroj, který bude použit při vytváření jednotlivých reportů, a další potřebné informace. Druhým krokem je zavedení funkcí, pomocí kterých se vytvoří konkrétní soubor, do aplikace.

```
private static HashMap<String, MessageDto> messages;

public HashMap<String, MessageDto> loadPageData(LinkedHashSet<String> set) {
	if (messages == null) {
		if(realPath.endsWith(".")) {
			realPath = realPath.substring(0, realPath.length() - 1);
		}
		String pathToXML = realPath + configuration.get("languageXML");

		messages = Utilities.parseXML(pathToXML);
	}
	HashMap<String, MessageDto> ret = new HashMap<String, MessageDto>();
	for (String key : set) {
		ret.put(key, messages.get(key));
	}
	return ret;
}
```

Ukázka kódu 4: Načítání jazykové lokalizace

Při psaní reportů bylo nutné vytvořit a k šabloně připojit scriptlet obsahující složitější funkce, které byly potřebné pro práci s daty v reportu. Scriptlet je v rámci reportů Java třída rozšiřující JRDefaultScriptlet, nebo implementující rozhraní JRAbstractScriptlet z JasperReports API. Tato třída poskytuje uživateli vhodný způsob, jak při navrhování šablony reportů používat vlastní potřebné funkce [10].

Při vytváření reportu na serveru probíhá načtení existující šablony, její naplnění potřebnými daty a následné uložení vytvořeného reportu. Takto uložený soubor již není problém vrátit zpět uživateli. Část funkce starající se o vytvoření reportu ukazuje zdrojový kód 5. Samotnou ukázku vytvořeného reportu s výpisem žádostí o dovolenou pak obrázek 3.

Poslední úkol byl, z důvodu složitého zakomponování reportovacích nástrojů do aplikace, časově nejnáročnější. Přesto po prvotním nezdaru, způsobeném špatnou kompatibilitou BIRT API s vytvářeným IS, byl díky pomoci zkušenějších kolegů zdárně dokončen během přibližně čtrnácti pracovních dnů.

```
JasperDesign jasperDesign;
JasperReport jasperReport;
JasperPrint jasperPrint;

//prepare design
filePath = realPath;
jasperDesign = JRXmlLoader.load(filePath+"/reports/" +
            (String) param.get("reportName") + ".jrxml");
jasperReport = JasperCompileManager.compileReport(jasperDesign);
jasperPrint = JasperFillManager.fillReport(jasperReport,param,
            connectionManager.createConnection());
Format formatter = new SimpleDateFormat("dd-mm-yyyy_HH-mm");
String fileName = param.get("reportName") + (String) param.get("abb_id") + "-" +
            formatter.format(new Date()) + (String) param.get("extension");
String filePathAndName = filePath+ "/reports/output/" + fileName;
String ext = (String) param.get("extension");

//render report to pdf
if(ext.matches(".pdf")) {
      JasperExportManager.exportReportToPdfFile(jasperPrint,filePathAndName);
}
```

Ukázka kódu 5: Vytvoření reportu

Náhradní volno

Příjmení, jméno, titul	Gargas Jiri				Osobní číslo	910305	
Costcentrum	C23033						
Žádá o dovolenou od	09.10.2012	do	16.10.2012	včetně, tj.	6	pracovních dnů	
Místo pobytu o dovolené							
Datum	03.11.2012	podpis zaměstnance					
Vyjádření vedoucího pracovníka:							
Datum		podpis vedouciho pracovníka					
Skutečný nástup dovolené		ukončení dovolené					
Datum		podpis odpovědného pracovníka					

Náhradní volno

Obrázek 4: Report náhradního volna

4 Závěr

4.1 Využité znalosti a dovednosti

V průběhu vykonávání bakalářské praxe jsem v první části velmi využil znalosti spojené s návrhem a implementací databázové struktury, získané v předmětech Úvod do databázových systémů a Databázové a Informační systémy. Při práci na jednotlivých úkolech jsem poté zužitkoval znalosti s objektovým programováním a znalosti z předmětu Programovací jazyky I, tedy základy jazyku Java. V průběhu vývoje jsem ocenil i zkušenosti z předmětu Softwarové inženýrství, které mi mnohdy pomohly velmi rychle pochopit zadanou práci, či jednoduše prezentovat mé nápady ohledně postupu zpracování jednotlivých úkolů. Nápomocná byla i schopnost hledat korektní informace, kterou studium vysoké školy velmi prohlubuje. Asi nejvíce jsem ale v průběhu celé praxe docenil všechny zadané semestrální projekty, a to proto, že většina z nich neměla pevné zadání a nutila mě tak využívat kreativní myšlení, které jsem při vývoji tohoto IS neustále potřeboval.

4.2 Chybějící znalosti a dovednosti

V začátcích vývoje IS jsem nejvíce postrádal zkušenosti s prací v týmu. S prací v týmu je spojena i znalost verzovacích nástrojů, které jsou opravdu potřebné, a se kterými jsem se setkal poprvé až právě během praxe. Při následném vývoji mi byla na obtíž i neznalost jednotlivých využívaných frameworků. Nicméně toto byla jen menší překážka, protože jsem o nich v průběhu studia již alespoň něco slyšel a jejich základy se daly pochopit velice rychle.

4.3 Dosažené výsledky a zhodnocení praxe

Během práce na projektu jsem přišel do styku s množstvím nových technologií, se kterými jsem si rád vyzkoušel pracovat. Jako příklad zde uvedu verzovací nástroj Mercurial, který jsem se naučil velmi dobře používat, a bez kterého si nyní již nedokážu představit žádny projekt. Zároveň jsme po celou dobu byli vedeni k dodržování konvencí psaní kódu, které jsem si tímto celkem osvojil. Získané zkušenosti upotřebím ještě během studia VŠ. Také jsem se poprvé podílel na vývoji IS ve skupině a zároveň na práci ve významné mezinárodní společnosti. Obě tyto zkušenosti jistě docením později, kdy mi, jak doufám, pomůžou projít zdárně sítem personalistů a získat tak dobrou pracovní pozici.

Samotnou praxi jsme začínali s prázdným projektem a seznamem funkcí, které by měl mít implementované. S odstupem času shledávám v našem postupu jedinou chybu. Tou byla volba frameworku GWT, ke kterému chybí kvalitní dokumentace a jehož komponenty mnohdy nefungují tak, jak popisuje dostupná dokumentace. I přes menší nesnáze, které nám GWT způsobil, jsme zdárně vytvořili IS, který splňuje všechny požadavky. Odbornou praxi tedy hodnotím z mého pohledu jako velmi přínosnou.

5 Reference

[1] ABB GROUP. The ABB Group - Automation and Power Technologies [online]. 2012 [cit. 16. 11. 2012]. Dostupné z: http://www.abb.com/

[2] ABB GROUP. ABB v České republice [online]. [cit. 10. 2. 2013]. Dostupné z: http://www.abb.cz/cawp/czabb014/6b8e288aa55b6384c1256b43005aeb0b.aspx

[3] GOOGLE INC. Google Web Toolkit [online]. [cit. 10. 11. 2012]. Dostupné z: https://developers.google.com/web-toolkit/

[4] MATT, Mackall. Mercurial SCM [online]. [cit. 10. 11. 2012]. Dostupné z: http://mercurial.selenic.com/

[5] JAVAFORGE. MercurialEclipse [online]. [cit. 10. 11. 2012]. Dostupné z: http://javaforge.com/project/HGE

[6] THE ECLIPSE FOUNDATION. EclipseLink [online]. [cit. 10. 11. 2012]. Dostupné z: http://www.eclipse.org/eclipselink/

[7] ORACLE. Java™ Platform, Standard Edition 7 API Specification [online]. [cit. 16. 11. 2012]. Dostupné z: http://docs.oracle.com/javase/7/docs/api/

[8] HTURKSOY, Hasant. FileUpload with GWT [online]. 2009 [cit. 2012-11-20]. Dostupné z: http://www.jroller.com/hasant/entry/fileupload_with_gwt

[9] HEROUT, Pavel. Java a XML. 1. vyd. České Budějovice: Kopp, 2007, 313 s. ISBN 978-80-7232-307-4.

[10] JASPERSOFT COMMUNITY. JasperReports Library [online]. 2012 [cit. 16. 11. 2012]. Dostupné z: http://community.jaspersoft.com/project/jasperreports-library